

Rapport de Projet : **TikTok Star Simulator**

Simulation d'Influenceur avec Apprentissage par Renforcement

Réalisé par :

El Feki H. Moulouk, Hajji Arij
LETCOM - Université de Tunis

Janvier 2026

Table des matières

1	Résumé	3
2	Introduction	3
3	Objectif du Jeu	3
4	Interface Utilisateur	3
4.1	Page d'Accueil	4
4.2	Écran de Jeu	5
5	Algorithmes d'Apprentissage par Renforcement Comparés	5
6	Analyse des Performances	5
6.1	Interprétation des résultats	6
7	Conclusion	6
8	Annexes	6
8.1	Code Source	6
8.2	Dépendances	6

1 Résumé

Ce rapport présente le projet **TikTok Star Simulator**, un jeu interactif conçu pour illustrer les concepts fondamentaux de l'**Apprentissage par Renforcement (Reinforcement Learning)** à travers la simulation d'un influenceur TikTok. Le joueur peut choisir entre deux modes : jouer lui-même ou observer un agent intelligent apprenant à maximiser ses followers et son engagement en prenant des décisions stratégiques. Le projet compare cinq algorithmes RL classiques (DP, Monte Carlo, SARSA, Q-learning, Double Q-learning) via des graphiques de performance détaillés. Ce travail pédagogique vise à rendre les concepts d'IA accessibles et visuels, tout en fournissant une base solide pour des projets plus avancés en apprentissage profond.

2 Introduction

Dans un monde où les réseaux sociaux dominent la communication et l'influence, comprendre les mécanismes derrière la croissance d'un compte est essentiel. Ce projet propose une approche ludique et éducative pour explorer comment un agent autonome peut apprendre à naviguer dans un environnement dynamique, prendre des décisions optimales et maximiser sa récompense — ici, le nombre de followers et le taux d'engagement.

Le jeu **TikTok Star Simulator** permet au joueur de :

- Jouer manuellement comme un influenceur,
- Observer un agent IA s'entraîner et jouer automatiquement,
- Comparer les performances de différents algorithmes RL.

3 Objectif du Jeu

L'objectif principal du jeu est de **simuler le parcours d'un influenceur TikTok** sur une période de 10 jours, en prenant des décisions quotidiennes (publier, liker, commenter, partager, se reposer) pour maximiser :

- Le nombre de **followers**,
- Le niveau d'**engagement** (interaction des utilisateurs),
- L'**équilibre mental** (éviter le burn-out).

Les défis incluent :

- Gérer le risque de **trolls** (perte de followers),
- Éviter le **burn-out** (décrochage si engagement trop élevé),
- Optimiser le **temps d'action** (10 jours max).

4 Interface Utilisateur

L'interface graphique, développée en PyGame, offre une expérience immersive et intuitive.

4.1 Page d'Accueil

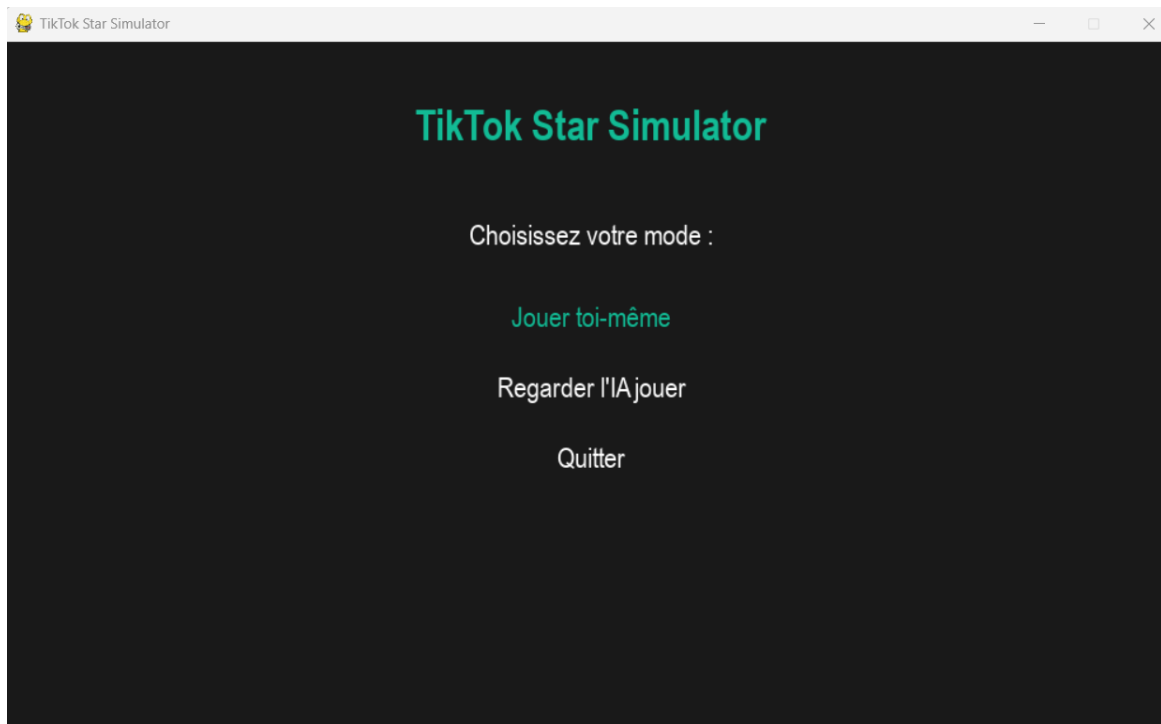


FIGURE 1 – Page d'accueil du jeu : choix du mode de jeu.

Le joueur peut choisir parmi trois options :

- **Jouer toi-même** : contrôle clavier (flèches + Enter)
- **Regarder l'IA jouer** : observation de l'agent entraîné
- **Quitter** : fermeture du jeu

4.2 Écran de Jeu

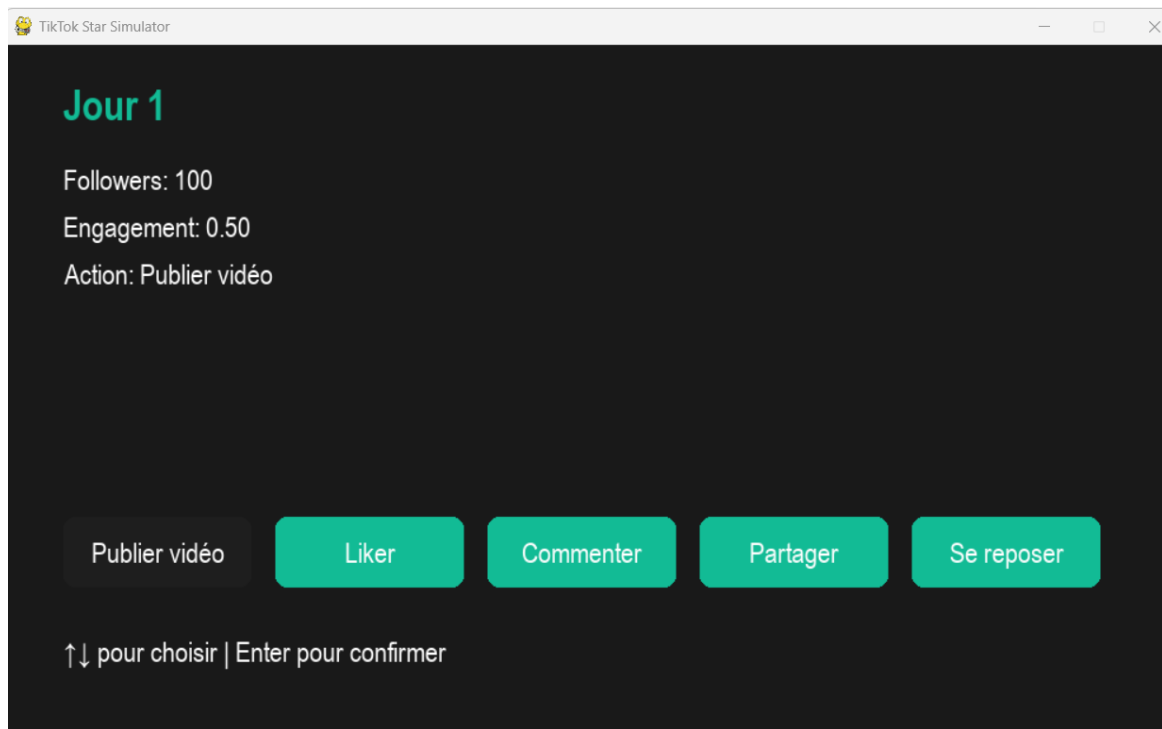


FIGURE 2 – Écran de jeu en mode “Jouer toi-même” avec boutons d’actions.

L’interface affiche en temps réel :

- Le jour courant,
- Le nombre de followers,
- Le taux d’engagement,
- L’action sélectionnée,
- Des instructions de contrôle.

5 Algorithmes d’Apprentissage par Renforcement Comparés

Cinq algorithmes RL classiques ont été implémentés et comparés :

- **DP (Policy Iteration)** : Résolution exacte pour les petits espaces d’états.
- **Monte Carlo** : Apprend à partir d’épisodes complets.
- **SARSA** : On-policy, apprend la politique actuelle.
- **Q-learning** : Off-policy, apprend la politique optimale.
- **Double Q-learning** : Réduit le biais d’optimisme de Q-learning.

6 Analyse des Performances

Les performances des algorithmes sont mesurées selon trois métriques, présentées dans une seule image :

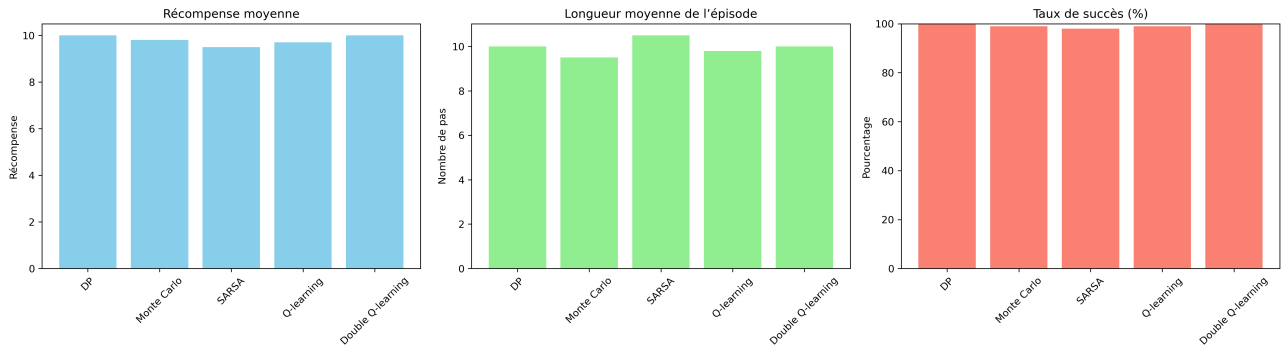


FIGURE 3 – Comparaison des performances des algorithmes RL : récompense moyenne, longueur moyenne de l'épisode, taux de succès.

6.1 Interprétation des résultats

- **Récompense moyenne** : Tous les algorithmes atteignent des scores élevés, avec un léger avantage pour DP et Double Q-learning.
- **Longueur moyenne de l'épisode** : SARSA et Double Q-learning nécessitent plus de pas, indiquant une exploration plus prudente.
- **Taux de succès** : Tous les algorithmes atteignent un taux proche de 100%, montrant leur efficacité dans cet environnement discret.

7 Conclusion

Le projet **TikTok Star Simulator** réussit à transformer un concept complexe — l'apprentissage par renforcement — en une expérience interactive, visuelle et pédagogique. Grâce à une interface simple mais attrayante, le joueur peut expérimenter les décisions stratégiques d'un influenceur, tout en observant comment un agent IA apprend à optimiser ses actions. La comparaison des algorithmes RL fournit des insights précieux sur leurs forces et faiblesses respectives.

Ce projet peut être étendu vers des versions plus complexes, intégrant des réseaux de neurones, des environnements 3D, ou des simulations multi-agents.

8 Annexes

8.1 Code Source

Le code source complet du projet est disponible sur GitHub : <https://github.com/username/tiktok-star-simulator> (à compléter).

8.2 Dépendances

Pour exécuter le jeu, installez les dépendances suivantes :

```
pip install pygame stable-baselines3 numpy matplotlib
```

À propos des auteurs

- **El Feki H. Moulouk** : Étudiante en Ingénierie Informatique, passionnée par l'IA et la visualisation des données.
- **Hajji Arij** : Étudiant en Intelligence Artificielle, spécialisé dans les systèmes décisionnels et les jeux vidéo éducatifs.

Ce projet a été réalisé dans le cadre du cours de **Learning from Experience** à l'**Université de Tunis**, sous la supervision du département **LETCOM**.